Raspberry Pi Based Computing Prototypes: Design, Implementation and Performance Analysis

Seth Wolfgang
School of Computing
Grand Valley State University
Allendale, Michigan, USA
wolfgans@mail.gvsu.edu

Xiang Cao
School of Computing
Grand Valley State University
Allendale, Michigan, USA
caox@gvsu.edu

2023 IEEE International Conference on Electro Information Technology (eIT) | 978-1-6654-9376-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/EIT57321.2023.10187384

***Abstract*—Nowadays, in Internet of Things (IoT) applications, IoT devices generate and gather data. These data can be handled in different ways. They can be processed by IoT devices directly, by the Cloud, or by the edge servers. In recent years, Raspberry Pi has been developed as a small IoT device for many applications. In this paper, we design and implement various computing models, i.e., Cloud Computing, Edge Computing, and computing on IoT devices, using Raspberry Pis as prototypes. We deploy multiple applications, run hands-on experiments, and analyze the results accordingly. We also compare the performances of these different computing paradigms, and take advantage of the parallel processing capability of Raspberry Pi devices to demonstrate the effectiveness of improving the prototype performance.**

***Index Terms*—Computing, Prototype, Raspberry Pi, Design, Implementation, Performance Analysis**

I. Introduction

In the Internet of Things (IoT) environment, huge amounts of data have been generated and collected by IoT devices. Traditionally, these data are directly transmitted to the Cloud for processing, because of the limited computing and storage capacities of IoT devices. Nowadays, with more powerful hardware, many IoT devices (e.g., smartphones) can process initial data and send results to the Cloud later.

Recently, a novel computing paradigm called edge computing [1] [2] has been popular. In the edge computing model, edge servers are deployed at locations that are closer to the data sources for better Quality of Service (QoS) performance, due to the short distance and reduced network latency. Thus, in edge computing, initial data collected by IoT devices are first sent to edge servers which further process them.

Hence, in today's IoT environment, three computing models are available, i.e., Cloud Computing (Cloud processes initial data), Edge Computing (Initial data are sent to edge servers), and computing on IoT devices (Initial data are processed by IoT devices).

In recent years, Raspberry Pi has been widely developed for many IoT applications as a small low-cost computing device. In addition to collecting data, Raspberry Pi can also store and process data as a server. In this paper, we design and implement Raspberry Pi based prototypes for IoT applications to demonstrate the performances among different computing paradigms (i.e., Cloud Computing, Edge Computing, and computing on IoT devices). We deploy multiple applications to conduct experiments and use total finishing time including data processing time and data transmission time as the performance metric.

In our work, Raspberry Pis are utilized for multiple purposes based on the computing paradigms they participate in. For example, in the Cloud Computing model, a Raspberry Pi device serves as a data collector, while in the models of Edge Computing and computing on IoT devices, it processes data. In many IoT applications, the processing and storage capabilities of Raspberry Pis can be effectively utilized to avoid wasting their otherwise unused computing power, due to their better and better hardware. We also take advantage of the fact that multiple Raspberry Pis can process data in parallel to improve the overall system performance.

We summarize the contributions of this paper as follows.

- We design and implement three computing prototypes based on Raspberry Pis, and demonstrate the performances of multiple IoT applications, in terms of the total finishing time.
- We analyze the experiment results, compare the performances among different computing paradigms, and discuss the advantage of each computing model in our prototype configurations.
- We utilize the computing power of Raspberry Pis and take advantage of their parallel processing capability to demonstrate the effectiveness of improving the overall prototype performance.

Compared with the previous work in [3], in this paper, we not only investigate edge computing, but also thoroughly compare and analyze the performances of IoT applications among different computing paradigms. We hope the results and conclusions of this paper can offer readers a better view of the performance of each computing paradigm and help the community understand the advantages and disadvantages of different computing models.

The rest of the paper is organized as follows. Section II is related work. We show our design, implementation, and prototype architectures in Section III. Section IV is the performance analysis. The conclusion and future work are in Section V.

978-1-6654-9376-5/23/$31.00 ©2023 IEEE

II. Related Work

Research work in [1] [2] provided general overviews of edge computing. They offered discussions of advantages, challenges, and what future work may hold about edge computing.

Some articles showcased their prototypes and systems used in their experiments about edge computing. The authors in [4] showed a prototype "EdgeAvatar" in which they created avatars using edge computing. In [5], the authors used edge computing to facilitate communications among vehicles, and created a vehicle edge server. Research work in [6] showed an edge computing prototype being used to control smart lighting on an airfield. In [7], the authors used edge computing to implement real-time surveillance, and demonstrated its feasibility and performance. The article in [8] presented an integrated system with edge computing about a satellite network for system performance improvement. These research work demonstrated the feasibility and popularity of edge computing among many applications.

Articles in [9] [10] [11] [12] utilized Raspberry Pis for edge computing research. The authors in [9] used Raspberry Pis to optimize distributed power flow for an electrical network. In [10], the authors deployed Raspberry Pis for facilitating an orchestration platform to help collect and analyze data at the edge level of their network. In [11], the authors created a parallel machine learning application for analyzing tourism data using a Raspberry Pi cluster. Research work in [12] implemented Raspberry Pi based clusters to create an edge cloud PaaS architecture, and showed its requirements could be met.

Other research work in [13] [14] [15] [16] [17] used Raspberry Pis to implement some interesting systems. In [13], the authors used a Raspberry Pi cluster to implement Hadoop MapReduce for data management of sensors. The authors in [14] showed an energy-efficient Raspberry Pi cluster for data mining applications to reduce power consumption. In [15], the authors took advantage of Raspberry Pis to replicate software and measured the performance. Research work in [16] demonstrated a prototype using a cluster of Raspberry Pis for MPI distributed computing experiments and discussed the performance. In [17], the authors demonstrated a network of smart home devices connected to a web server. Raspberry Pis were used in their system.

Different from the previous work in [3], our work in this paper offers a more comprehensive comparison among different computing paradigms including computing on IoT devices. Also, this paper implements more applications to thoroughly demonstrate the performance of multiple computing models. Our design architectures and conclusions in this paper are not limited to Raspberry Pi devices, but also they can be applied to various computing paradigms using other devices.

III. Design, Implementation and Prototype Architectures

A. Performance Metric: Total Finishing Time of Applications

As an important metric to evaluate the performances among different computing models to deal with data, in this paper, we use the total finishing time of applications. Same as in [3], the total finishing time is defined as the sum of data processing time plus the data transmission time. In other words, the total finishing time is the time for the Cloud, edge server, or IoT device to process data, plus the time taken by the network to transmit data. From the view of Quality of Service (QoS), it is best to have a reduced total finishing time, so that the Cloud can obtain the result/conclusion quickly.

B. Computing Models

In IoT applications, data are generated and gathered by IoT devices. There are multiple models to process/compute these data, i.e., Cloud Computing, Edge Computing, and computing on IoT devices.

- In the Cloud Computing model, the IoT device gathers data and directly sends them to the Cloud for further processing to get the result/conclusion. In this model, it is fast to process data since the Cloud has a powerful computing capability. However, due to the long distance between the IoT device and the Cloud, the time to transmit the raw data can be large.
- In the Edge Computing model, the IoT device first sends the raw data to the edge server, which is usually deployed at a close location. The edge server then processes the data and sends the result/conclusion to the Cloud. In this model, the processing time could be larger, since the computing capability of the edge server could be less powerful than the Cloud. However, the data transmission can be reduced, because only the result/conclusion is sent to the Cloud.
- In the model of computing on IoT devices, the IoT device directly processes the raw data it collects, then sends the result/conclusion to the Cloud. In this model, the processing time could be large, due to the limited computing capability of the IoT device. However, the data transmission could be small, since only the result/conclusion is sent out.

In this paper, we explore the tradeoff of performance among different computing models. We use various applications to demonstrate the experiment results.

C. Prototype Architectures

In this paper, we use Raspberry Pis as both an edge server and IoT device to emulate different computing models. The fact that Raspberry Pi is less powerful than a regular computer, is similar to the comparison between edge server and the Cloud. Also, because of its small size, a Raspberry Pi is often utilized as an IoT device to collect data.

Figure 1 shows our prototype structures. We use a regular laptop to emulate the Cloud as it is more powerful than Raspberry Pi. To make a difference in performance between the edge server and IoT device, we use a Raspberry Pi 4 as the edge server, whereas a Raspberry Pi 3B+ is used as the IoT device. In order to simulate network conditions for different computing models, we intentionally use wired or wireless connections based on specific scenarios.

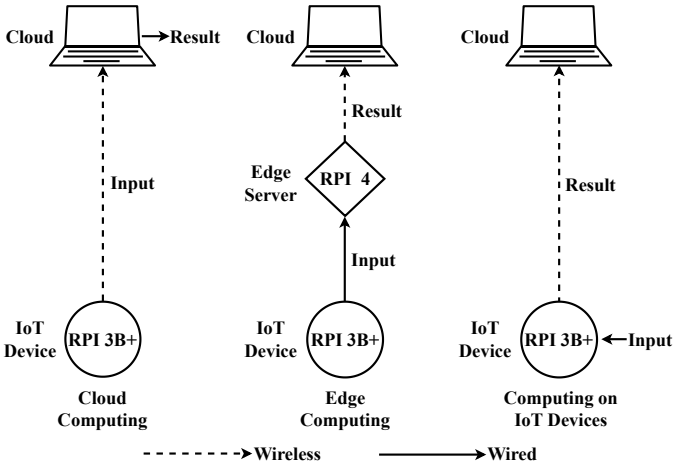

Fig. 1. Prototype Structures

In the Cloud Computing model shown in Figure 1, the Cloud directly processes the input data sent from the IoT device, and generates the result/conclusion. We use a wireless link to connect the IoT device and the Cloud together, simulating a long distance and a large network delay between them. In the Edge Computing model, a wired connection is used for the IoT device to send the input data to the edge server, indicating the edge server is deployed at the location close to the IoT device with a small network delay. Also, the edge server processes the input data. Since the Cloud is far away from the edge server with a large network delay, we use a wireless connection to send the result/conclusion to the Cloud. In the model of computing on IoT devices, the IoT device directly processes the input data and sends the result/conclusion to the Cloud using a wireless connection.

D. Applications

In this paper, we use various applications to conduct hands-on experiments to show the performances of different computing models. Since the data processing capability of the Cloud is more powerful than edge server and the IoT device, we use the following criteria to choose our applications to give models of Edge Computing and computing on IoT devices a fair opportunity to compete with Cloud Computing, same as the work in [3].

- The size of the raw data (input data) is large, so it takes time to transmit them through the network.
- The size of the result/conclusion (output data) is small, so they can be quickly sent out to the Cloud.

With these criteria, the models of Edge Computing and computing on the IoT device can avoid transmitting a large amount of raw (input) data to the Cloud. Instead, only the small-size results/conclusions (output data) are sent out. Hence, the data transmission times of Edge Computing and computing on IoT devices can be reduced, giving these two computing models a chance to compensate for the longer data processing times (due to their less powerful processing capabilities) compared with the Cloud Computing model.

We implement the following three applications.

1) Tesseract for Optical Character Recognition:

Tesseract [18] is an application program for Optical Character Recognition (OCR). The input data of Tesseract is an image and its output data is a string of characters recognized by the program as the result/conclusion.

2) Smith-Waterman Gene Sequence Alignment:

The Smith-Waterman gene sequence alignment [19] [20] is the second application we adopt in this paper. Its input data are multiple gene string files and the output data is a number showing the length of a matched sequence of these files. In this application, the similarity score is generated based on a matrix as the result/conclusion.

3) Logistic Regression:

Logistic regression [21] takes in multiple data points as the input data, and creates an equation that best fits all data given as the result/conclusion.

We can see that all three applications above follow our criteria that the output data (result/conclusion) sizes are much smaller than the sizes of input data, so that the output data can be quickly transmitted via the network.

E. Computing Power of Raspberry Pis

Raspberry Pis are developed as small low-cost computing devices. Due to their better and better hardware, the computing power of Raspberry Pis become a valuable resource. In this paper, we believe a Raspberry Pi not only can be an IoT device to collect data (in Cloud Computing model), but also it can provide data processing capability (as in the models of Edge Computing and computing on IoT devices). Hence, we hope the computing resources of Raspberry Pis can be fully utilized in many IoT applications.

We also consider and take advantage of the parallel processing offered by Raspberry Pis. With multiple Raspberry Pis running in parallel, the overall system performance can be improved in some cases.

IV. Performance Analysis

A. Experiment Setup

In this paper, we use a Samsung Galaxy Book Pro laptop featuring an i7-1165G7 processor, 16 GB of RAM, and Windows 11, as the Cloud. In our experiments, the Raspberry Pi 4 (Model B) with 8GB of RAM is used as an edge server and the Raspberry Pis 3B+ with 1GB of RAM is used as an IoT device. Raspbian OS is the operating system for these Raspberry Pi devices. As mentioned previously, the total finishing time of applications is the performance metric. We also show the data processing time and data transmission time as the results.

B. Our Results

1) Tesseract for Optical Character Recognition:

We run the Tesseract (OCR) application [18] with small-size and large-size input data respectively. In each set of experiments, we repeat running the same data in 25, 50 and 75 iterations so that the difference of results among various computing models can be clearly demonstrated.

TABLE I
TESSERACT OCR, SMALL-SIZE INPUT

Computing Model	Processing Time (Sec)	Transmission Time (Sec)	Total Finishing Time (Sec)
25 Iterations			
Cloud	8.1533	0.8023	8.9556
Edge	35.3608	0.0011	35.3619
IoT Device	92.3905	0.0273	92.4179
50 Iterations			
Cloud	16.2710	1.3021	17.5731
Edge	68.9588	0.0015	68.9604
IoT Device	184.5172	0.0280	184.5451
75 Iterations			
Cloud	24.3007	2.0346	26.3353
Edge	103.3494	0.0034	103.3528
IoT Device	276.5449	0.0370	276.5819

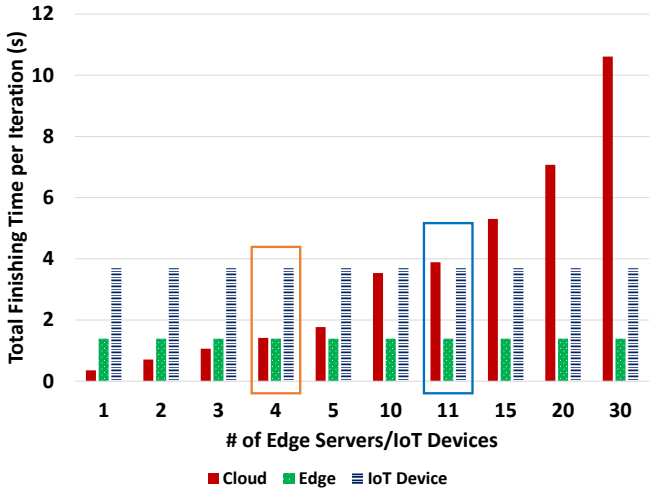

Fig. 2. Projected Performance: Tesseract OCR, Small-size Input

- Small-size Input:

  The input data for this set of experiments is a sample Michigan license plate. Its resolution is 1300 x 687 pixels, and the size is 104 KB.

  Table I shows the experiment result. We can see that for all iterations, the processing time for the Cloud is the shortest since it has the best data processing capability. The Edge Computing has a shorter processing time than the IoT device, because the hardware (Raspberry Pi 4) of the edge server is better than that of the IoT device (Raspberry Pi 3B+).

  In terms of the data transmission time, the Edge Computing produces the smallest result, whereas the Cloud Computing model has the longest time. This is because the entire raw input image data is sent from the IoT device to the Cloud in the Cloud Computing model, enduring a long delay. On the contrary, in the model of computing on IoT devices, only the result/conclusion is sent to the Cloud, greatly reducing the data amount. The Edge Computing model is the best, because in addition to the fact that only the result/conclusion is sent, its hardware (Raspberry Pi 4) is also better than the IoT device (Raspberry Pi 3B+), so the data can be sent quicker.

  Due to our budget limit, we only have a limited number of Raspberry Pis, thus without being able to further demonstrate the performance with more Raspberry Pis. Therefore, we use the average total finishing time per iteration for all cases shown in Table I to estimate the performance in a large scale deployment with more edge servers/IoT devices. In other words, the average total finishing time per iteration is the metric for performance projection.

  Figure 2 shows the projected performance for all three computing models in small-size input data. With only one edge server and one IoT device, the Cloud Computing greatly outperforms the other two computing models, due to its more powerful data processing capability. With more and more edge servers and IoT devices, the performance of Edge Computing and computing on IoT devices remain the same, because they can process and transmit data in parallel. However, the performance of Cloud Computing becomes worse and worse, because it deals with data sequentially. We can see that starting from 4 edge servers and IoT devices, the Edge Computing has already outperformed the Cloud, whereas with 11 of them, the model of computing on IoT devices provides better performance than the Cloud. After that, the Cloud Computing has the longest total finishing time.

- Large-size Input:

  In this set of experiments, the input data is an image containing a string of characters ”GRAND VALLEY STATE UNIVERSITY”. Its resolution is 2992*2992 pixels, and the size is 1.1MB. Table II demonstrates the experiment result. It shows that the processing time and transmission time for all iterations are larger than those in small-size input data for all computing models, as the input data sizes become larger. Following the same trend of small-size input data, results in large-size input also show that the Cloud outperforms the Edge Computing and computing on IoT devices in terms of the processing time. Also, for the data transmission time, the Edge Computing model is the best one, whereas the Cloud Computing has the worst result, due to the same reason mentioned in the discussion of the small-size input result.

  Figure 3 shows the projected performance in large-size input data. We can see that similar to the result of small-size input, with more and more Edge Servers and IoT devices, the performances of Edge Computing and computing on IoT devices gradually outperform the Cloud Computing, due to the advantage of parallel processing. Further compared with Figure 2, the result in Figure 3 demonstrates that the Edge Computing and computing on IoT devices need a smaller number of (i.e., 3 and 8 respectively) edge servers/IoT devices to catch up with the performance of the Cloud. This is because the Edge

TABLE II
TESSERACT OCR, LARGE-SIZE INPUT

Computing Model	Processing Time (Sec)	Transmission Time (Sec)	Total Finishing Time (Sec)
25 Iterations			
Cloud	16.8065	2.3868	19.1932
Edge	53.2056	0.0187	53.2243
IoT Device	147.0599	0.0502	147.1101
50 Iterations			
Cloud	33.4197	4.5953	38.0149
Edge	106.4486	0.0241	106.4727
IoT Device	295.4589	0.0776	295.5364
75 Iterations			
Cloud	53.6931	6.8867	60.5798
Edge	159.5799	0.0417	159.6216
IoT Device	439.7309	0.1098	439.8406

TABLE III
SMITH-WATERMAN, SMALL-SIZE INPUT

Computing Model	Processing Time (Sec)	Transmission Time (Sec)	Total Finishing Time (Sec)
100 Iterations			
Cloud	15.1611	12.9940	28.1551
Edge	141.4989	0.0311	141.5300
IoT Device	211.2504	0.0777	211.3281
200 Iterations			
Cloud	22.0721	16.9003	38.9724
Edge	286.0233	0.0679	286.0912
IoT Device	420.6392	0.1774	420.8166
300 Iterations			
Cloud	33.1990	30.8396	64.0386
Edge	423.6244	0.0992	423.7236
IoT Device	632.5821	0.2324	632.8145

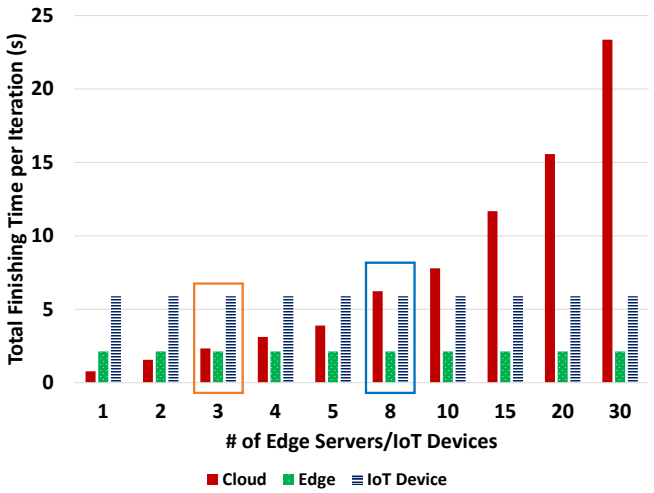

Fig. 3. Projected Performance: Tesseract OCR, Large-size Input

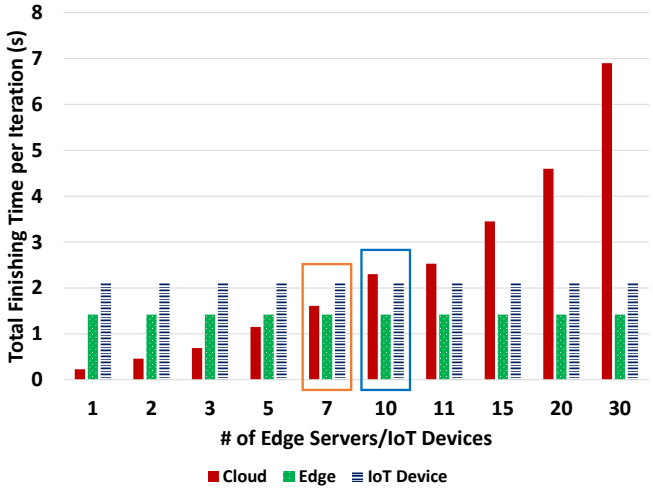

Fig. 4. Projected Performance: Smith-Waterman, Small-size Input

Computing and computing on IoT devices in large-size input provide relatively better performance (compared with the Cloud) than that in small-size input. Hence, the advantage of parallel processing (leading to smaller total finishing time) can be reflected earlier.

2) Smith-Waterman for Similarity Score of Genes:

The experiments of the Smith-Waterman application [20] are conducted in small-size and large-size input data as well. In each set of experiments, we run 100, 200, and 300 iterations respectively. This application are based on four files "alphabet.txt", "scoringmatrix.txt", "database.txt" and "query.txt" in [20]. The actual inputs are data selected from them. The first two files "alphabet.txt" and "scoringmatrix.txt" are used in both small-size and large-size input experiments. The "alphabet.txt" is a simple four-byte text file with the characters "ACGT" listed. The "scoringmatrix.txt" is a 4x4 matrix with positive ones on the diagonal, and negative ones everywhere else. The file sizes of "database.txt" and "query.txt" are different for small-size and large-size input experiments.

- Small-size Input:

  The actual sizes of input files for this set of experiments are 49.1KB (selected from "database.txt") and 92 bytes (selected from "query.txt"). From the results shown in Table III, we can see that the Cloud still outperforms the other two computing models for the processing time for all iterations, whereas computing on IoT devices produces the worst result. In terms of the data transmission time, the Cloud has the worst performance, which is significantly longer than those of the other two computing models. Compared with the input data of the Tesseract OCR application, with more files and more iterations, the total input data sizes of the Smith-Waterman application become larger, leading to longer data transmission time of the Cloud Computing, so that its data transmission time accounts for a much larger portion of the total finishing time. On the contrary, for the other computing models, data processing still takes the most time. Due to the same reason mentioned previously, the Edge Computing has the best performance for the data transmission time.

  Figure 4 demonstrates the result of projected performance in small-size input data. We can see that with the advantage of parallel processing, Edge Computing and

TABLE IV
SMITH-WATERMAN, LARGE-SIZE INPUT

Computing Model	Processing Time (Sec)	Transmission Time (Sec)	Total Finishing Time (Sec)
100 Iterations			
Cloud	38.7618	14.3556	53.1174
Edge	575.7464	0.0365	575.7828
IoT Device	837.8275	0.0866	837.9141
200 Iterations			
Cloud	59.4789	20.4252	79.9040
Edge	1151.9231	0.0742	1151.9972
IoT Device	1686.7155	0.1910	1686.9064
300 Iterations			
Cloud	89.4748	39.1918	128.6665
Edge	1722.6685	0.1166	1722.7851
IoT Device	2526.7014	0.2456	2526.9470

TABLE V
LOGISTIC REGRESSION, SMALL-SIZE INPUT

Computing Model	Processing Time (Sec)	Transmission Time (Sec)	Total Finishing Time (Sec)
100 Iterations			
Cloud	1.5198	5.9881	7.5080
Edge	7.5300	0.0225	7.5524
IoT Device	21.8093	0.1145	21.9237
200 Iterations			
Cloud	3.0020	13.1244	16.1264
Edge	14.9240	0.1980	15.1220
IoT Device	44.2514	0.2969	44.5483
300 Iterations			
Cloud	4.4833	16.8687	21.3519
Edge	22.3877	0.2202	22.6079
IoT Device	66.2403	0.4374	66.6778

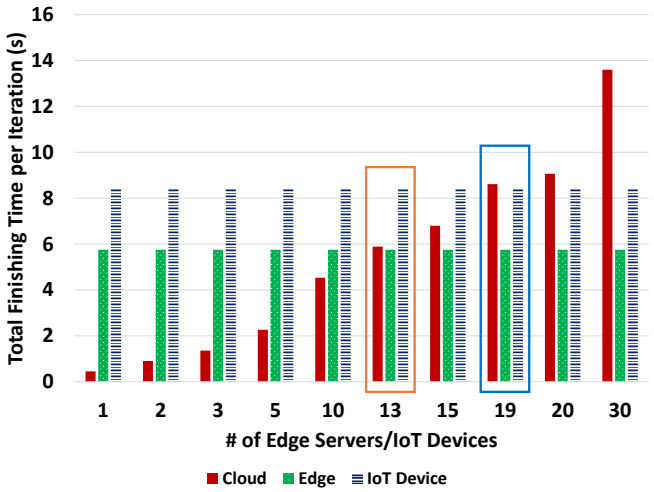

Fig. 5. Projected Performance: Smith-Waterman, Large-size Input

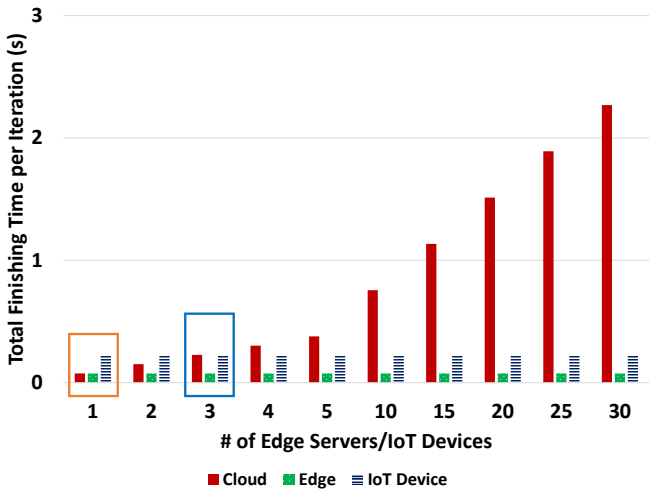

Fig. 6. Projected Performance: Logistic Regression, Small-size Input

computing on IoT devices have better performance starting from 7 and 10 edge servers/IoT devices respectively. Compared with the results in Figure 2 and Figure 3, the other two computing models need more edge servers/IoT devices to catch up with the performance of the Cloud. This is because the total finishing times of the other two computing models in this set of experiments are relatively longer (compared with the Cloud) than those in Tables I and II, due to the characteristic of the input data.

- Large-size Input:

  The sizes of input files for this set of experiments are 174KB (selected from "database.txt") and 137 bytes (selected from "query.txt"). Table IV shows the result. We have the same conclusion as the small-size input (i.e., the Cloud has the best performance for the processing time, but its data transmission time is the longest one among all computing models. The data transmission time of the Cloud Computing is still a relatively much larger portion of the total finishing time, compared with the other two computing models. Also, the Edge Computing provides the shortest data transmission time for all iterations.)

  Figure 5 shows the projected performance in large-size input. We can see the same conclusion that the Cloud provides worse and worse performance when the number of edge servers/IoT devices increases. Also, compared with the result shown in Figure 4, the Edge Computing and computing on IoT devices need more edge servers/IoT devices (13 and 19 respectively) to outperform the Cloud. This is because the total finishing times of the other two computing models in large-size data input are relatively larger (compared with the Cloud Computing) than those in small-size data input.

3) Logistic Regression:

Same as the Smith-Waterman application, we run the application of logistic regression [21] in small-size and large-size input data respectively. In each set of experiments, 100, 200, and 300 iterations are executed to make the results large enough. The inputs of this application are data selected from two files "BreastCancer.txt" and "testData.txt" in [21].

- Small-size Input:

  In this set of experiments, the input data sizes are 200KB for both input files. The result in Table V still shows that the Cloud Computing model produces the

TABLE VI
LOGISTIC REGRESSION, LARGE-SIZE INPUT

Computing Model	Processing Time (Sec)	Transmission Time (Sec)	Total Finishing Time (Sec)
100 Iterations			
Cloud	2.9758	9.7352	12.7110
Edge	19.9277	0.0476	19.9753
IoT Device	50.8672	0.1530	51.0202
200 Iterations			
Cloud	5.9089	17.8599	23.7688
Edge	27.0008	0.2042	27.2049
IoT Device	101.7555	0.5798	102.3353
300 Iterations			
Cloud	8.8565	17.9006	26.7570
Edge	60.4337	0.6673	61.1010
IoT Device	137.0968	0.7129	137.8097

best result for the processing time, whereas its data transmission time is the worst one. It is also worth noting that in this application, the data transmission time of the Cloud Computing is larger than the data processing time, whereas for the other two computing models, the data processing times are still the dominant parts. The Edge Computing model has the best data transmission time for all iterations.

Figure 6 shows the projected performance in small-size input. We can see that due to its relatively much larger data transmission time than the data processing time, the Cloud Computing has already had longer total finishing time than the Edge Computing, even with only one edge server. Also, due to the relatively larger total finishing time (compared with the other two computing models) produced by the Cloud than in the Tesseract OCR and Smith-Waterman applications, with only 3 edge servers/IoT devices, the performance of the Cloud has been caught up by the model of computing on IoT devices.

- Large-size Input:

  In this set of experiments, the input data sizes are 399KB for both input files. The result shown in Table VI follows the same trend of the small-size input experiments mentioned above. Hence, it verifies our idea that the Cloud Computing does well in processing data due to its powerful computing capability. However, due to a long delay from the IoT device, the Cloud suffers the largest data transmission time. Same as the result of the small-size input experiments, the data transmission time is larger than the data processing time in the Cloud Computing model. Also, the Edge Computing can provide the shortest data transmission time, due to its hardware advantage compared with the IoT device and the fact that only the result/conclusion is sent.

  Figure 7 shows the projected performance in large-size input. Due to the characteristic of input data, compared with the other two computing models, the Cloud Computing has relatively better performance in total finishing time than the result in the small-size input experiments.

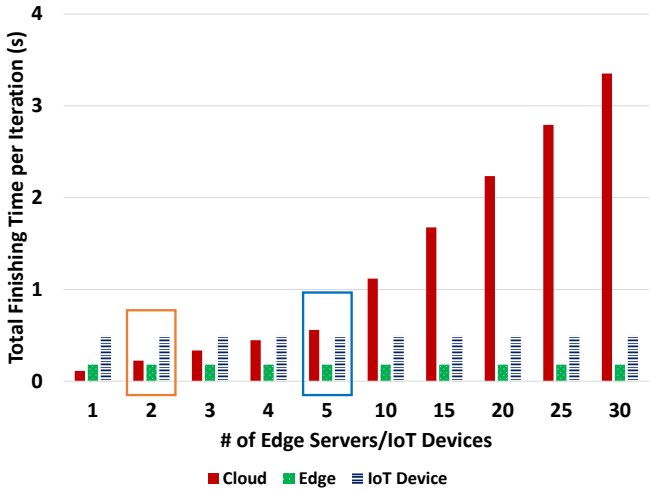

Fig. 7. Projected Performance: Logistic Regression, Large-size Input

Hence, Edge Computing and computing IoT devices need more edge servers/IoT devices (2 and 5 respectively) to outperform the Cloud Computing.

C. Key Takeaways

Based on results and projected performance of all three applications, here are our key takeways.

- With only a single edge server/IoT device, the Cloud Computing always provides the best performance in terms of the data processing time than the other two computing models, due to its most powerful processing capability. However, the Cloud produces the largest data transmission time, because the raw input data are sent from the IoT device with long delays.
- The Edge Computing has the smallest data transmission time among all three computing models. This is because the edge server has better hardware than the IoT device and only the result/conclusion is sent to the Cloud.
- For a large-scale deployment, parallel processing offered by multiple edge servers and IoT devices can provide better performance than the Cloud for the total finishing time. With enough edge servers, the Edge Computing has the best result.
- The results from various applications in small and large input data demonstrate that Edge Computing and computing on IoT devices need different number of edge servers/IoT devices to catch up with the performance of the Cloud Computing.

V. CONCLUSION AND FUTURE WORK

In many IoT applications, there are three computing models to deal with data, Cloud Computing, Edge Computing, and computing on IoT devices. In this paper, we design Raspberry Pi based prototypes and implement these different computing models with various applications. We analyze the experiment results, and compare the performance among these computing paradigms. We also take advantage of the parallel processing

provided by Raspberry Pis to demonstrate the performance improvement. In the future work, we plan to further explore our prototypes in a larger scale using more Raspberry Pi devices, and investigate the impact on the performance.

REFERENCES

[1] W. Shi, G. Pallis and Z. Xu, "Edge Computing [Scanning the Issue]," in Proceedings of the IEEE, vol. 107, no. 8, pp. 1474–1481, Aug. 2019.

[2] F. Liu, G. Tang, Y. Li, Z. Cai, X. Zhang and T. Zhou, "A Survey on Edge Computing Systems and Tools," in Proceedings of the IEEE, vol. 107, no. 8, pp. 1537–1562, Aug. 2019.

[3] T. Gizinski and X. Cao, "Design, Implementation and Performance of an Edge Computing Prototype Using Raspberry Pis," in Proceeding of the 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC), 2022, pp. 0592–0601.

[4] N. Watkinson, F. Zaitsev, A. Shivam, M. Demirev, M. Heddes, T. Givargis, A. Nicolau, and A. Veidenbaum, "EdgeAvatar: An Edge Computing System for Building Virtual Beings," Electronics, vol. 10, no. 3, p. 229, Jan. 2021.

[5] G. S. Pannu, J. L. L. Altet, T. Higuchi, S. Ucar, O. Altintas and F. Dressler, "Demo: Making It Real - Virtual Edge Computing in a 3D Driving Simulator," 2019 IEEE Vehicular Networking Conference (VNC), 2019, pp. 1–2.

[6] A. Mijuskovic, R. Bemthuis, A. Aldea and P. Havinga, "An Enterprise Architecture based on Cloud, Fog and Edge Computing for an Airfield Lighting Management System," in 2020 IEEE 24th International Enterprise Distributed Object Computing Workshop (EDOCW), 2020, pp. 63–73.

[7] J. Wang, J. Pan, and F. Esposito, "Elastic urban video surveillance system using edge computing," in Proceedings of the Workshop on Smart Internet of Things (SmartIoT '17), ACM, Article 7, 1—6.

[8] C. Li, Y. Zhang, R. Xie, X. Hao and T. Huang, "Integrating Edge Computing into Low Earth Orbit Satellite Networks: Architecture and Prototype," in IEEE Access, vol. 9, pp. 39126–39137, 2021.

[9] D. Gebbran, G. Verbič, A. C. Chapman, and S. Mhanna, "Coordination of Prosumer Agents via Distributed Optimal Power Flow: An Edge Computing Hardware Prototype," in Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS '20), 2104-–2106.

[10] L. Miori, J. Sanin, and S. Helmer, "A Platform for Edge Computing Based on Raspberry Pi Clusters," in: Calì A., Wood P., Martin N., Poulovassilis A. (eds) Data Analytics, BICOD 2017, Lecture Notes in Computer Science, vol 10365, Springer, Cham.

[11] A. Komninos, I. Simou, N. Gkorgkolis, and J. Garofalakis, "Performance of Raspberry Pi microclusters for Edge Machine Learning in Tourism," in Edge Machine Learning For Smart Iot Environments Workshop (Edging), 2019 European Conference On Ambient Intelligence (Ami2019), Rome, Italy.

[12] C. Pahl, S. Helmer, L. Miori, J. Sanin and B. Lee, "A Container-Based Edge Cloud PaaS Architecture Based on Raspberry Pi Clusters," in 2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), 2016, pp. 117–124.

[13] H. Karchalkar and P. Railkar, "Raspberry Pi Hadoop Cluster based Data Processing," in IJCA Proceedings on International Conference on Internet of Things, Next Generation Networks and Cloud Computing ICINC 2016(2):1–3, July 2016.

[14] J. Saffran et al, "A Low-Cost Energy-Efficient Raspberry Pi Cluster for Data Mining Algorithms," in: Desprez F. et al. (eds) Euro-Par 2016: Parallel Processing Workshops, Lecture Notes in Computer Science, vol 10104, Springer, Cham.

[15] H. Knoche and H. Eichelberger, "Using the Raspberry Pi and Docker for Replicable Performance Experiments: Experience Paper," in Proceedings of the 2018 ACM/SPEC International Conference on Performance Engineering (ICPE '18), 305-–316.

[16] K. N. Myint, M. H. Zaw, and W. T. Aung, "Parallel and Distributed Computing Using MPI on Raspberry Pi Cluster," in International Journal of Future Computer and Communication vol. 9, no. 1, pp. 18–22, 2020.

[17] S. Sagar, U. Choudhary and R. Dwivedi, "Smart Home Automation Using IoT and Raspberry Pi," in Proceedings of the International Conference on Innovative Computing & Communications (ICICC), 2020.

[18] Tess4J, "A Java JNA wrapper for Tesseract OCR API," [Online]. Available: https://tess4j.sourceforge.net/

[19] "Smith–Waterman algorithm," in Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Smith-Waterman_algorithm

[20] JayakrishnaThota, "Sequence-Alignment," GitHub repository, [Online]. Available: https://github.com/JayakrishnaThota/Sequence-Alignment

[21] Agelos369, "Logistic-Regression-JAVA," GitHub repository, [Online]. Available: https://github.com/Agelos369/Logistic-Regression-JAVA